Patterns for Maude Metalanguage Applications

Georgiana Goriac
Eugen-Ioan Goriac

Sinaia School on Formal Verification of Software Systems
3-10 March 2008
http://www.imar.ro/~diacon/sinaiaschool.html

Topics

- Software patterns
- Maude metalanguage applications
- Case study: a topological sorting system

Software patterns

- introduced by Christopher Alexander
  - urban design and building architecture
- common language used in order to describe :
  - a design problem
  - a context in which the problem occurs
  - the core of a solution to solve the problem

The problem

- specifying and analizing a system
- system examples:
  - simulators
  - provers
  - models of computation

Maude metalanguage applications

- a particular type of application in which Maude is used to define modules for specifying:
  - a language syntax
  - a language parser
  - a way of execution
  - a manner of printing execution results

Case study – the TOPO system

```
Maude>
(poset SIMPLE-POSET is
  rel a < b .
  rel e < b .
  rel b < c .
 end)
```

- special syntax for defining a partial order set

```
Maude> tsort c d a b e .
```

- call of a topological sorting command

```
result: a d e b c .
```

- system response

Deeper analysis

- User Interface
  - define the communication flow between the user and the system under implementation
- System Language Signature
  - define the system language signature used in order to validate system inputs
- System Language Parser
  - develop a parser in Full Maude for transforming the input matching the system language grammar into a semantics in terms of the Maude language

User Interface

- system loop mode

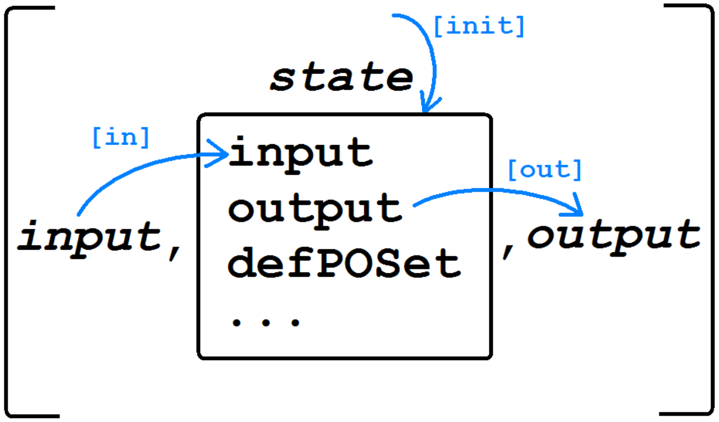
  ```
  [input:QidList, state:State, output:QidList]
  ```

- system state structure
  - an object characterized by attributes

    ```
    input : TermList
    output : QidList
    defPOSet : Header ...
    ```

- user interface rewrite rules

  ```
  [init], [in], [out]
  ```



Sinaia School on Formal Verification of
Software Systems, 3-10 March 2008

System Language Signature

- TOPO grammar

```
POSet ::= poset Name is Relation* end
Name ::= Identifier
Relation ::= rel LHS < RHS .
LHS ::= Obj
RHS ::= Obj
Obj ::= a | b | ... | z
```

- declaration of metavariable sorts

```
sorts @POSet@ @Relation@ .
```

- declaration of metaexpressions corresponding operators

```
op poset_is_end : @Token@ List{@Relation@} -> @POSet@ .
op rel_<_. : @Token@ @Token@ -> @Relation@ .
op tsort_. : @Bubble@ -> @Command@ .
```

System Language Parser

- the association of Maude semantics to the user input
- example:

```
(poset ORDER is     (mod ORDER is
                       including BOOL .
                ⇨      including ITEMS .
   rel a < b .         eq a < b = true .
 end)                 endm)
```

- steps:
  - creating an operator for parsing some input
  - creating a rule that calls the parsing operator

System Language Parser

```
op parsePOSet : Term Term -> Module .
eq parsePOSet(T, T’) = ... --- make use of the metaParse operation

crl [parseUnit-POSet] :
      < O : X@Database | db : DB,
        input  : ('poset_is_end[T, T']),
        output : nil,
        Atts
      >
      =>
      < O : X@Database | db : insTermModule(getName(M), M, DB),
        input  : nilTermList,
        output : ('\n 'Introduced 'poset 'specification: getName(M) '\n),
        Atts
      >
      if M := parsePOSet(T, T') .
```

Applying the patterns

- Maude metalanguage applications can be developed by using an iteration-based strategy
- The idea is to build the base version of the system to be implemented and then, at each iteration to add new capabilities to that system
- Every time an iteration is performed, the enriched system has to be tested for errors

About the patterns

- The design of these patterns is based on the experience acquired by the authors during the development of some applications or by studying other applications
- The greatest achievement is the refactoring of the CIRC proving tool, based on the patterns

References

- Eugen-Ioan Goriac, Georgiana Caltais, Dorel Lucanu, Oana Andrei and Gheorghe Grigoras
  Patterns for Maude Metalanguage Applications
  (accepted at WRLA'08, to appear in ENTCS)

- http://circidei.info.uaic.ro/pmma2008/topo.maude

- http://www.imar.ro/~diacon/sinaiaschool.html